CSCE 6352 Human-Computer Interaction
Requirements
Brian Gauch
Due: Thursday, September 24, 2015

I pledge to pursue all academic endeavors with honor and integrity. I understand the principles of the Honor System, and I promise to uphold these standards by adhering to the Honor Code in order to preserve the integrity of Vanderbilt University and its individual members.

Mission Statement:

Goal:

The proposed system should allow for a more efficient, filtered search for teammates in the game League of Legends (LoL) as compared to in-game methods.

Scope:

The scope of this project encompasses providing ways for users to register League of Legends (LoL) teams that are seeking players. It will allow other users to search for openings and use email to notify successful applicants. It will be guaranteed to work only on Windows platforms with the Chrome browser. The planned system will make use of a publically available LoL API to gather data about team members to facilitate recommending teams to users. If that API turns out not to have the needed information, I will create a synthetic data set for testing. The system will not verify that a user owns the linked LoL account. The system will not request team information from the LoL API (e.g. current team of a user).

User Profiles:

Timmy is new to LoL and is looking for a fun group to play with. He wouldn't mind if one or more players on his new team were significantly better than him, because then he could learn from them. He is not sure how often he will end up playing, or when.
-motivation: low
-frequency of use: once
-skill level with system: low (he is young, so the system would need to be simple)
-task nature: Find a team

Spike is an experienced player of games similar to LoL, but has only recently switched to LoL (which is why he hasn't found a team yet). He is looking for a similarly experienced team, because he doesn't want to be held back by any weaker players. He plays almost every day after work.
-motivation: medium
-frequency of use: once
-skill level with system: medium (uses similar web pages but has not used the system)
-task nature: Find a team

Lily has played a fair amount of LoL but only in "solo queue", where she was automatically matched with a different allied team each game. That has gotten a bit old, so she wants to try playing with a team. She doesn't care very much about how often the team wins; she just wants to see what kind of crazy stuff they can do. She only plays on weekends.
-motivation: low
-frequency of use: once

-skill level with system: medium (uses similar web pages but has not used the system)
-task nature: Find a team

Alex and three members of his team are looking for a replacement, because they could rarely all be online at the same time as their former 5th member, who was in another time zone. They mainly play on weekdays after school.
-motivation: high (needs 5th member to play with other friends)
-frequency of use: once
-skill level with system: medium (uses similar web pages but has not used the system)
-task nature: Find a teammate

Inputs/Outputs:

Inputs:

From user:

- Their LoL username
- Their intent (Case A: Find a team OR Case B: Find new team members)
- Case A:
  - Their selection of team
- Case B:
  - Team name and description

From LoL:

- Current ranks of users
- Matchmaking histories (times of matches only)

Outputs:

Informs users of relevant teams
Informs teams when a user wishes to join
Informs users when a team accepts them

Constraints:

Environmental:

-Because LoL is a PC game that requires a keyboard and mouse, it is expected that most potential users of the proposed system will be using a PC. So, I don't think we need to worry too much about small screen sizes or bad lighting.

Computational:

-We are limited in how much data we can request from LoL's API. Also, there are a large number of League of Legends players. We will therefore have to be selective about what we request and when.

-Can't easily verify that a user owns a LoL account.

Actual Requirements:

Requirement #: 1.0
Requirement Name: Team account creation
Requirement Type: Functional
Description: The system needs to be able to create an account when a user registers a new team with the system. The team name and password need to be checked for uniqueness and recorded. The creator must be a valid LoL player, and is automatically placed on the team. The user will be notified if their username or password does not fit the requirements.
Rationale: Teams need to be able to manage their openings and view applicants. Others need to be prevented from manipulating their data.
Variable Ranges: Valid team names are 6 or more characters long and alphanumeric.
Valid passwords are 6 or more characters long and alphanumeric.
Evaluation: We will evaluate by trying valid and invalid team usernames and passwords, as well as previously registered team names. We will test with a valid and an invalid LoL username as the team creator.
Dependencies: 7.0 (although we can use a synthetic dataset if needed), 8.0, 9.0.
Conflicts: None

Requirement #: 2.0
Requirement Name: Team account management
Requirement Type: Functional
Description: When a user is logged in on behalf of a team, they should be able to view and/or modify the team profile, i.e. the team members and description. Team member names provided must match LoL usernames.
Rationale: Teams need to be able to provide up-to-date information to potential applicants.
Variable Ranges: LoL teams have 1-5 members. Therefore, the number of openings for a team plus the number of current members of a team should sum to 1-5. The number of openings must be in the range 1-4.
Evaluation: We will evaluate by trying to enter valid and invalid numbers of openings.
Dependencies: 1.0, 7.0
Conflicts: None

Requirement #: 3.0
Requirement Name: Team opening management
Requirement Type: Functional
Description: When a user is logged in on behalf of a team, they should be able to view and/or modify the number of openings.
Rationale:
Variable Ranges: Teams (both in LoL and our system) have a maximum size of 5 members.
Evaluation: We will evaluate by trying valid and invalid usernames, and descriptions that are too long.
Dependencies: 1.0, 7.0
Conflicts: None

Requirement #: 3.1
Requirement Name: Posting an opening
Requirement Type: Functional
Description: When a user is logged in on behalf of a team, they should be able to view and/or modify the desired number of additional teammates.
Rationale: Teams should not be matched with potential new members if they do not desire any new members.
Variable Ranges: LoL teams have 1-5 members. Therefore, current team size must be 1-4 in order to post an opening.
Evaluation: We will evaluate by trying valid and invalid numbers of openings.
Dependencies: 1.0
Conflicts: None

Requirement #: 3.2
Requirement Name: Closing an opening
Requirement Type: Functional
Description: When a user is logged in on behalf of a team, they should be able to view and/or modify the desired number of additional teammates.
Rationale: Teams should not be matched with potential new members if they do not desire any new members.
Variable Ranges: The current number of openings must be 1-4.
Evaluation: We will evaluate by trying valid and invalid numbers of openings.

Dependencies: 1.0
Conflicts: None

Requirement #: 4.0
Requirement Name: Team review of applicants
Requirement Type: Functional
Description: When a user is logged in on behalf of a team, they should be able to select among applicants, if any. This decision should be informed.
Rationale: Both the new member and the team must consent to the new member's team membership.
Variable Ranges: Team must have a 1 or more opening.
Evaluation: We will evaluate by trying valid and invalid usernames, and descriptions that are too long.
Dependencies: 1.0, 6.0
Conflicts: None

Requirement #: 4.1
Requirement Name: Team accept an applicant
Requirement Type: Functional
Description: When a user is logged in on behalf of a team, they should be able to reject applicants that they do not think would make good team members. It is expected that, afterwards, outside our system, the team members and applicant will friend each other in-game and/or form an in-game team. They will be able to do this because our system provides their counterpart(s) LoL username(s). If the last opening is filled, all other applicants are notified of automatic rejection and removed from the team's queue of applicants.
Rationale: Both the new member and the team must consent to the new member's team membership.
Variable Ranges: Team must have a 1 or more opening.
Evaluation: We will evaluate by making sure that the user receives the acceptance message and the application is removed from the team's queue of applicants. We will also verify that all applications are removed from the queue if this filled the last opening. In that case, affected applicants must be notified of automatic rejection.
Dependencies: 4.0
Conflicts: None

Requirement #: 4.2
Requirement Name: Team reject an applicant
Requirement Type: Functional
Description: When a user is logged in on behalf of a team, they should be able to reject applicants that they do not think would make good team members.
Rationale: Both the new member and the team must consent to the new member's team membership.
Variable Ranges: None
Evaluation: We will evaluate by making sure that the user receives the rejection message and the application is removed from the team's queue of applicants.
Dependencies: 4.0
Conflicts: None

Requirement #: 5.0
Requirement Name: User search for team
Requirement Type: Functional

Description: When a user enters a LoL username in the search box, a list of recommended teams is provided. The user will select a rank option. These results will be ranked by opening age, ascending or descending, or alphabetically. If time allows, we will also add ranking based on preferred play times or skill level.
Rationale: Users want to see teams with openings.
Variable Ranges: Choices of ranking option
Evaluation: We will evaluate by trying valid and invalid usernames, and we will test each of the ranking options.
Dependencies: 1.0, 2.0, 3.1, 7.0, 8.0, 9.0
Conflicts: None

Requirement #: 5.1
Requirement Name: User examines team profile
Requirement Type: Functional
Description: When a user clicks on a search result, they see more detailed information about the team, including team members and description.
Rationale: Users need more information about teams before they decide which to join.
Variable Ranges: NA
Evaluation: We will select a search result to view detailed team information
Dependencies: 1.0, 2.0, 5.0
Conflicts: None

Requirement #: 6.0
Requirement Name: User applies to join team
Requirement Type: Functional
Description: When a user is viewing a team that has at least one opening, they may request membership. This should be a single input.
Rationale: Users want to join teams.
Variable Ranges: User must not already be on the team.
Evaluation: We will apply as a user that is already a member of the team, and as a non-member user.
Dependencies: 5.1
Conflicts: None

Requirement #: 6.1
Requirement Name: User is notified of acceptance or rejection
Requirement Type: Functional
Description: User is notified via email of acceptance or rejection after a team has made a decision about their application.
Rationale: Users should be made aware of the status of their application.
Variable Ranges: NA
Evaluation: We will test with accepted and rejected user applications.
Dependencies: 4.1, 4.2, 6.0
Conflicts: None

Requirement #: 7.0
Requirement Name: Database of valid LoL usernames
Requirement Type: Non-Functional

Description: We need access to a database of valid LoL usernames to validate applicants and team members. For advanced ranking algorithms, we also need information about users preferred playing times and skill levels.
Rationale: We need to be sure that teams consist of actual LoL users.
Variable Ranges: NA
Evaluation: We will test getting information out of the LoL API. If we can't get the data we need, we will create a synthetic dataset with at least 100 users.
Dependencies: None
Conflicts: None

Requirement #: 8.0
Requirement Name: System must be accessible from a web browser
Requirement Type: Environmental
Description: User must be able to use the system on a Chrome browser on a Windows machine.
Rationale: LoL is available on Windows or OSX. I don't have time to code for all browsers.
Variable Ranges: NA
Evaluation: We will test our system on Windows with Chrome.
Dependencies: NA
Conflicts: None

Requirement #: 9.0
Requirement Name: LoL players only
Requirement Type: User Requirements
Description: Either to create a team or to join a team, a user must enter a valid LoL username.
Rationale: Only LoL players can be on LoL teams.
Variable Ranges: NA
Evaluation: NA
Dependencies: None
Conflicts: None

Requirement #: 10.0
Requirement Name: Ease of use
Requirement Type: Usability Requirements
Description: The system should be easily understandable and useable without any training, and with few user errors.
Rationale: Since this is only for a game, users will not put in the effort to learn how to use a complicated system.
Variable Ranges:
Evaluation: We will conduct a thorough user study.
Dependencies: None
Conflicts: None

Requirement #: 10.1
Requirement Name: Contextual help
Requirement Type: Usability Requirements
Description: The system should provide information on valid data ranges and error recovery.
Rationale: Users should not be penalized for making an error, and should instead be able to complete their task promptly.

Variable Ranges: NA
Evaluation: We will conduct a thorough user study.
Dependencies: None
Conflicts: None

Proposed Schedule

September 10 – 23	Develop Code to Access Data from the API
September 24	Requirements Project Due
September 24 – 30	Develop Web Interface with Menus/Options*
	Develop Team Management Code
October 1	Prototyping Plan Due
October 1-12	Develop User Search Code
	Develop Team Display Code
	Develop Openings Management Code
October 13	Prototype Demonstration
October 12 – 21	Alpha Test with 2-3 stakeholders
	Modify Prototype based on Alpha Testing
	Recruit 10-20 Volunteer Stakeholders for Experiment
October 22	User Test Plan Due
October 24	User Testing
November 30	Final Project Due

*Moved later based on feedback on the Project Plan
** Other minor renamings of tasks based on requirements